Midterm 2012

1 Formalizmus (10 bodov) – zadanie

V tejto úlohe kladťte hlavný dôraz na presnosť a korektnosť práce s formalizmom z prednášky.

V tejto úlohe je povolené bez dôkazu použiť len nasledujúce skutočnosti:

- Definíciu množiny primitívne rekurzívnych funkcií,
- V každej podúlohe znenia predchádzajúcich podúloh.
- Primitívnu rekurzívnosť funkcie $mul(x, y) = x \cdot y$.

Vyriešte nasledujúce úlohy:

a) (1 bod) Dokážte, že každá konštantná funkcia s aritou 0 je primitívne rekurzívna.

b) (3 body) Dokážte, že každá konštantná funkcia s aritou 1 je primitívne rekurzívna.

c) (2 body) Dokážte, že úplne každá konštantná funkcia je primitívne rekurzívna.

d) (4 body) Dokážte, že ak je f unárna primitívne rekurzívna funkcia, tak je aj g primitívne rekurzívna.

$$\forall n : g(n) = \prod_{x<n} f(x)$$

(Špeciálne $g(0)$ je prázdny súčin, teda $g(0) = 1$.)

2 Formalizmus (10 bodov) – riešenie

a) Nulárnu nulu máme z definície: je ňou funkcia z. Ostatné konštanty vyrobíme postupnou kompozíciou so successorom: položme $f_0 \equiv z$ a pre $i > 0$ funkciu f_i vyrobíme ako $Comp[s, f_{i-1}]$.

b) Unárnu nulu vyrobíme primitívnou rekurziou z nulárnej, napr. nasledovne: $z_1 \equiv PR[z, P_1^2]$. Ostatné unárne funkcie vyrobíme zo z_1 kompozíciou so successorom, rovnako ako v a).

c) Konštantnú nulu s aritou n vieme vyrobiť zo z_1 napr. kompozíciou s vhodnou projekciou: $z_n \equiv Comp[z_1, P_1^n]$. No a ostatné konštanty s aritou n vyrobíme zo z_n osvedčeným postupom.

d) Funkcia g spĺňa nasledovnú rekurzívnu definíciu: $g(0) = 1$ a $\forall n : g(n+1) = g(n) \cdot f(n)$. Podľa nej môžeme g vyrobiť primitívnou rekurziou. Základný prípad nám zabezpečí konštantná funkcia s aritou 0, vracajúca hodnotu 1. (Tú sme si vyrobili v časti a), označíme si ju j.)

Ako druhú funkciu na výrobu g potrebujeme funkciu, ktorá z hodnôt $g(n)$ a n vyrobí hodnotu $g(n+1)$. Táto funkcia vyzerá tak, že na druhý argument použije f a výsledok vynásobí s prvým argumentom. Najskôr si teda zostrojíme funkciu f_2, ktorá dostane dva argumenty a na druhý z nich použije f: bude $f_2 \equiv Comp[f, P_2^2]$. Následne z tejto f_2 vyrobíme funkciu r nasledovne: $r \equiv Comp[mul, P_1^2, f_2]$.

Funkcia g teraz vzniká primitívnou rekurziou z funkcií j a r. Po dosadení dostávame:

$$g \equiv PR[\ \underbrace{Comp[s, z]}_{j},\ \underbrace{Comp[mul, P_1^2, Comp[f, P_2^2]]}_{r}\ ].$$

3 Známa pôda (5+5 bodov) – zadanie

V tejto úlohe sa pýtam na priamo odprednášané veci, ukážte, že im dostatočne rozumiete.

- Funkcia $\varphi : \mathbb{N}^2 \to \mathbb{N}$ rýchlo rastie. Až tak rýchlo, že ku každej unárnej primitívne rekurzívnej funkcii f existuje konštanta q_f taká, že $\forall n \in \mathbb{N} : \varphi(n, q_f) > f(n)$.
- Funkcia $\psi : \mathbb{N}^2 \to \mathbb{N}$ vie všetko, čo vedia unárne primitívne rekurzívne funkcie: ku každej unárnej primitívne rekurzívnej funkcii f existuje konštanta q_f taká, že $\forall n \in \mathbb{N} : \psi(q_f, n) = f(n)$.

Môže byť φ alebo ψ primitívne rekurzívna? Rozhodnite a dokážte.

4 Známa pôda (5+5 bodov) – riešenie

Funkcia φ nemôže byť primitívne rekurzívna. Sporom. Keby bola, bola by primitívne rekurzívna aj unárna funkcia g definovaná predpisom $g(n) = \varphi(n, n)$. Potom by ale funkcia g musela byť väčšia sama od seba, čo je spor.

Podrobnejšie: ak je g primitívne rekurzívna, tak (podľa zadania) existuje konštanta q_g taká, že $\forall n \in \mathbb{N} : \varphi(n, q_g) > g(n)$. Keďže to platí pre všetky n, platí to aj pre $n = q_g$. Teda $g(q_g) < \varphi(q_g, q_g)$. Ale zároveň podľa definície g platí $g(q_g) = \varphi(q_g, q_g)$, a to je hľadaný spor.

O funkcii ψ to môžeme dokázať podobne. Ale stačí si uvedomiť, že ak by existovala primitívne rekurzívna ψ, tak funkcia ξ definovaná $\xi(a, b) = 1 + \psi(b, a)$ je tiež primitívne rekurzívna a zároveň spĺňa podmienky pre funkciu φ.

5 Exkurzia do neznáma (10 bodov) – zadanie

V tejto úlohe sa pýtam na niečo, čo som neprednášal, a chcem vidieť, ako si s tým poradíte.

Uvažujme nasledovný model vypočítateľnosti:

- Program je konečná postupnosť inštrukcií (očíslovaných 0 až $n - 1$).
- Stroj má sedem premenných, nazvaných a až g. Premenné môžu obsahovať ľubovoľné nezáporné celé čísla.
- Stroj počíta unárnu funkciu. Na začiatku je v a vstup, v ostatných premenných sú nuly. Na konci má byť v b výstup, v ostatných premenných môže byť čokoľvek. Vykonávanie programu začína inštrukciou 0.
- Inštrukcie sú troch rôznych typov:
  `x:=x+1` zväčší premennú x o 1
  `if x>0 then x:=0 else goto i` ak je premenná x kladná, vynuluj ju, inak pokračuj inštrukciou i
  `if x=y then goto i` ak premenné x a y obsahujú tú istú hodnotu, pokračuj inštrukciou i
- Ak sa nevykoná goto, stroj sa snaží pokračovať nasledujúcou inštrukciou. Ak tá už neexistuje, alebo sa vykoná goto na číslo neexistujúcej inštrukcie, výpočet končí.

Rozhodnite a dokážte, či je tento model vypočítateľnosti Turingovsky úplný.
Záchrana za 3 body: ak neviete úlohu vôbec riešiť, aspoň napíšte program počítajúci identitu.

6 Exkurzia do neznáma (10 bodov) – riešenie

Najskôr program počítajúci identitu:

```
0. if a=b then goto 3
1. b:=b+1
2. if g=g then goto 0
```

Slovne: zvyšujem b, kým nenastane rovnosť a a b. Všimnite si použitie `if g=g` ako podmienky, ktorá je vždy splnená. (A ak sa niekomu nepáči, že neporovnávam dve rôzne premenné, rovnako dobre sme mohli použiť `if f=g`.)

A teraz dokážeme, že je Turingovsky úplný. Na to je potrebné ukázať *dve* tvrdenia:
– že má *aspoň* takú výpočtovú silu ako Turingovsky úplné modely,
– že má *nanajvýš* takú výpočtovú silu ako Turingovsky úplné modely.

To druhé tvrdenie je síce v tomto prípade pomerne očividné, aj tak som však v písomke chcel o ňom vidieť aspoň zmienku. (Kto tak nespravil, mal o pár bodov menej.) Existujú totiž aj modely, ktoré sú od Turingovsky úplných ostro silnejšie, a teda treba ukázať, že tento náš medzi ne nepatrí. Na tento dôkaz si stačí všimnúť, že máme inštrukcie sformulované tak, že predstavujú korektné príkazy v Pascale. A už vieme, že náš zjednodušený Pascal (ktorý obsahuje tieto príkazy a navyše aj nejaké iné) je Turingovsky úplný. Každý program v našom novom programovacom jazyku teda môžeme odsimulovať vhodným Pascalovským programom.

Ostáva teda ťažšie tvrdenie. Na jeho dôkaz si potrebujeme vybrať nejaký Turingovsky úplný model vypočítateľnosti a ukázať, že vieme ľubovoľný program v ňom odsimulovať. Intuitívne dobrou voľbou budú Minského registrové stroje s 2 registrami.

Aby sme vedeli odsimulovať ľubovoľný program pre Minského registrový stroj, stačí vedieť odsimulovať jednotlivé inštrukcie. Hodnoty registrov budeme mať v premenných a a b. Zväčšiť premennú vieme – takú inštrukciu má priamo aj náš stroj. Aj test na nulu je ľahký – necháme si v premennej g stále 0 a s ňou budeme porovnávať premennú, ktorú chceme otestovať.

Ostáva teda zmenšenie hodnoty v registri, ak je nenulová. BUNV nech chceme zmenšiť hodnotu v registri a. To spravíme vhodnou postupnosťou inštrukcií: Otestujeme, či je a nulová. Ak áno, nič nerobíme. Ak nie, inicializujeme c na 1 a d na 0. Kým sa c nerovná a, zvyšujeme súčasne aj c, aj d. A keď nastane $a = c$, tak v d máme hodnotu $a - 1$. Vynulujeme a a zvyšujeme ho dovtedy, kým sa nerovná d.

Program:

```
x+ 0.  if a=g then goto x+12
x+ 1.  if c>0 then c:=0 else goto x+2
x+ 2.  c:=c+1
x+ 3.  if d>0 then d:=0 else goto x+4
x+ 4.  if a=c then goto x+8
x+ 5.  c:=c+1
x+ 6.  d:=d+1
x+ 7.  if g=g then goto x+4
x+ 8.  if a>0 then a:=0 else goto x+9
x+ 9.  if a=d then goto x+12
x+10.  a:=a+1
x+11.  if g=g then goto x+9
```